To integrate Chainlink with Ethereum and come up with its advantages over Ethereum

Kshitij Garg
Department of Computer Science and Information System
BITS Pilani, Pilani, India
f20200120@pilani.bits-pilani.ac.in

***Abstract*—A Blockchain is a transparent, immutable ledger shared across nodes which stores transactions and helps in tracking assets. Thus eliminating any third party which might act as a single point of failure. Smart contracts or self executing contracts, do just this by allowing all this functionality to be written in the form of code and be executed on the blockchain when called upon. They eliminate the need for a middleman or any trust between the parties to carry a transaction. The capabilities of smart contracts are however, limited by the lack of the presence of live or off chain data on the blockchain. This gap between the on-chain and the off-chain world is filled with the help of oracles, which act as bridges between the two worlds. Chainlink is one of the most promising solutions to the 'Oracle Problem'. It offers a plethora of Decentralized Oracle Networks (DONs) to address the needs of blockchain developers in different aspects of the Oracle problem, while maintaining complete decentralization. One such revolutionary service offered by Chainlink is the Oracle Computation Service, which will be discussed in further detail with special reference to its application in mobile crowd sensing.**

***Index Terms*—Ethereum, Chainlink, Oracles, Oracle Computation, Mobile Crowd Sensing.**

I. Introduction

A Blockchain is a transparent, immutable ledger shared across nodes which stores transactions and helps in tracking assets. Thus eliminating any third party which might act as a single point of failure [1]. While the tracking of assets is open to everyone this does not mean that the privacy of the network is compromised. This is because every user is hidden behind a public address, hence as long as the address cannot be linked to a person, the transactions remain anonymous in a sense. As the 'ledger' is immutable, a node is only allowed to add data and not change previously added data. This is achieved by creating consensus between nodes, which do not necessarily need to trust each other. Things work out as long as the participants trust the system through which they achieved consensus. This consensus can be achieved using a number of popular consensus protocols, the first of which was proposed by Satoshi Nakamoto in his Bitcoin whitepaper [2].

The appropriate consensus protocol varies with the use case and the nature of the blockchain such as private, public or consortium. The various types of consensus protocols solve the 51% attack problem in various ways:
1.PoW(Proof of Work)
2.PoS(Proof of Stake)
3.DPos(Delegated Proof of Stake)
4.Proof of Space
5.Proof of Elapsed Time
6.Proof of Authority (Used by Rinkeby Testnet) [3]

As mentioned earlier, Blockchains being a decentralised system remove the need for middlemen to carry out tasks,enforce contracts and handle conflicts. Smart contracts or self executing contracts, do just this by allowing all this functionality to be written in the form of code and be executed on the blockchain when called upon. They eliminate the need for a middleman or any trust between the parties to carry a transaction. Smart contracts are a type of 'account' i.e. they have a balance and an address and can therefore carry out transactions. They are executed over the Ethereum Virtual Machine. [4] Smart contracts can be written using languages like Solidity or Vyper. An excellent IDE for the same would be Remix. Deploying a smart contract is similar to any other transaction and requires the payment of a fee or gas. Smart contracts can be thought of as open APIs.

The one major limitation of this revolutionary idea is that they themselves have no access to real world information/data because they cannot send HTTP requests. This is by design. Relying on external information could jeopardise consensus, which is important for security and decentralization. This causes a huge loss in adaptability and hence a loss in their application to the real world. Blockchains are deterministic by design-they execute exactly as written with a much higher degree of certainty than traditional systems, thus integration of an oracle like functionality into the base layer was avoided on purpose. Blockchains are not well suited to answer questions that are subjective, require external data that may not be equally accessible to all nodes or data that may vary with time. By being purposely isolated from external systems, Blockchains obtain their most valuable properties like strong consensus on the validity of user transactions, prevention of double-spending attacks, and reduction of network downtime. However 'Oracles' provide a workable solution. A blockchain oracle is a secure piece of middle ware that facilitates communication between blockchains and any off-chain system, including data providers, web APIs, enterprise back-ends, cloud providers, IoT devices, e-signatures, payment systems, other blockchains, and more. In short, oracles are a trusted third party source of information that give real-world data to a blockchain. [5]

But for the system to work, the input cannot come from a single or centralized source, because this would go against the very nature of a blockchain. This is the very problem that Chainlink helps solve. [5]

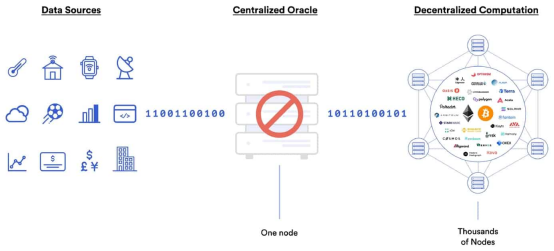

Fig. 1: The Oracle Problem

Chainlink is a decentralised proof-of-stake network of nodes that connects off-blockchain data and information to on-blockchain smart contracts via oracles. In order to bring determinism to the oracle layer, Chainlink has developed a network of decentralized oracle networks (DONs), with each DON involving a combination of multiple security techniques needed to service a particular use case. A Decentralized Oracle Network, or DON for short, combines multiple independent oracle node operators and multiple reliable data sources to establish end-to-end decentralization.

Another limitation of smart contracts is the largely demotivating gas price. Keeping in mind the limited computational power of the blockchain in comparison to traditional networks it becomes increasingly prohibitive to carry out large computations on the chain, coupled with the maximum contract size of 24KB for smart contracts.

Chainlink not only offers a method to deliver off chain data but also a revolutionary service called as 'Oracle Computation'. It provides a method of moving complex gas intensive computations off-chain, which are verified using chainlink's extensive network of oracles and makes the result available on-chain. [6]

II. Background

A chainlink request is pretty similar to an API call, with the added advantage that it can be made from within a smart contract. Financial smart contracts, for example, require market data to make settlement decisions, insurance smart contracts require IoT and web data to make payout decisions, trade finance smart contracts require trade documents and digital signatures to know when to release payments, and plenty of smart contracts want to settle in fiat currency on a standard payment network. Chainlink provides access of all available off-chain data to smart contracts which can be used in a variety of applications.

When a smart contract puts out a request for some off-chain data, it gets registered as an event by the Chainlink protocol. A Service Level Agreement Contract is generated on the blockchain, which in turn generates three sub-contracts, namely, a Chainlink Reputation Contract, a Chainlink Order-Matching Contract, and a Chainlink Aggregating Contract. The Chanlink Core is used to translate requests into off chain language and the fetched results into on-chain language. [7]

Reputation Contract

It is responsible for checking the oracle provider's past record for verifying its authenticity and reliability. The contract discards disreputable or unreliable nodes.

Order-Matching Contract

The order Matching Contract delivers the requesting contract request to chainlink nodes and allows them to bid for the request in case no specific set of nodes has already been chosen by the requesting contract. It then selects the right number and types of nodes to fulfil the request.

Aggregating Contract

The aggregating contract aggregates the data returned by the chosen oracles and validates it for accurate results. It can reconcile data from single or multiple sources. The Chainlink Aggregating Contract can repeat this validation process for multiple sources and then reconcile all validated data by averaging it into a single piece of data.

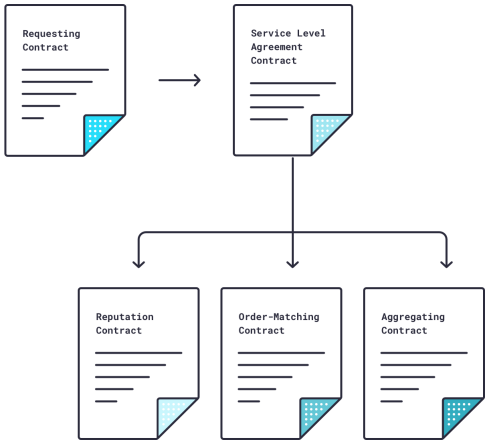

Fig. 2: Types of Chainlink Contracts [8]

Now in an application like mobile crowd sensing, whereby a large group of individuals having mobile devices acting as sensors and capable of computing collectively, share data and extract information to measure, map, analyze, estimate or infer any processes of common interest, the presence of off-chain data becomes invaluable. Moreover, this large mass of data needs to be validated before being used. This validation process in itself is a computationally heavy task, which coupled with high gas prices would drive the cost of the operation to enormous heights.

The data to be fetched was taken as weather data collected by a public API by the name 'openweathermap.org' [9], which would further need to be extended to a custom build public authenticated API fetching data from the remote sensors. Eventually the data would also need to be stored on some decentralized data storage system like IPFS [10], from where it could be fetched using its unique key and on the basis of key parameters like date, location and sensorID. The fetching of

this custom data and its off-chain computation was attempted as part of the implementation aspect of the project.

III. Related Work

Huang et al. in their paper titled "Blockchain Based Mobile Crowd Sensing in Industrial Systems" [11] proposed integrating the mobile crowd sensing based smart industrial systems with a blockchain layer. They identified that traditional Mobile Crowd Sensing systems are vulnerable to a variety of malicious attacks and have a magnitude of single point of failure due to the presence of a centralized architecture. However the data generated by the mobile sensors is highly unreliable unless it has been properly validated and is stored on a decentralized data storage. While the above paper moves on to provide an optimized Mobile Crowd Sensing model using blockchains, the presence of on-chain validation makes the process increasingly non-profitable. In general creating a service which might demands its users to do the heavy-lifting is also not commercially viable. Hence Off-chain or Oracle computation seems to be a great way out of the problem. Moreover connecting sensor nodes onto a centralized network defeats the very purpose of decentralizing the application layer using blockchains, thus fetching off-chain and also cross-chain (say from a decentralized storage like IPFS) data also becomes increasingly necessary.

Chen et al. in their paper titled "On blockchain integration into mobile crowd sensing via smart embedded devices: A comprehensive survey" [12] come up with similar problems, which in my opinion can be well catered using Chainlink services like keepers, off-chain computation and off-chain reporting.

Liu et al. in their paper titled "Off-chain Data Fetching Architecture for Ethereum Smart Contract" [13] highlight the importance of oracle based data fetching for decentralized applications and also explain the basic structure of an oracle service.

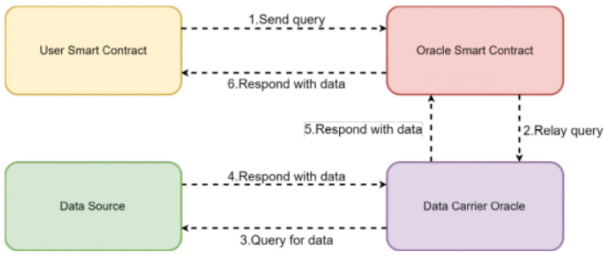

Fig. 3: Structure of an Oracle [13]

IV. Proposed Methodology/Solution/Design

Chainlink provides a number of services to enhance the privacy and scalability of Ethereum and similar blockchains. Some of the most prommising services include [14]:

Random Number Generator(RNG)

Chainlink's Verifiable Randomness Function is a safe and provably fair Random Number Generator (RNG) that creates on-chain cryptographic proofs that verify the randomness wasn't tampered with. The application of true random numbers are endless especially within the gaming industry and potentially in the metaverse. When users access Portals, Aavegotchi, an on-chain gaming project on the Polygon sidechain, uses Chainlink VRF to swiftly and effectively mint provably unique Aavegotchi NFTs with randomly determined features. [15]

Privacy Preserving Data Queries and Credential Management

The use of DECO (Decentralized Oracles) service proposed by Zhang et al. in their paper tilted "DECO: Liberating Web Data Using Decentralized Oracles for TLS" [16] allows for all HTTPS/TLS data to be confidentially attested by oracles without revealing the data on the chain or the actual identity of the person to whom the data corresponds.

Keepers

Smart contracts remain passive by default and need to be called upon to perform certain functions. Keepers provide a decentralized network of nodes capable of automating smart contract functioning and upkeep of the smart contract. An example can be a smart contract that needs to execute at exactly 5 A.M. IST everyday. Therefore to access this off-chain data of the current time in IST and call upon the smart contract can be done using Chainlink Keepers.

Chainlink offers two types of adapters namely Core and External adapters.Core adapters are functionalities that come inherently installed with the chainlink core node client software.Chainlink External Adapters enable easy integration of custom computations and specialized APIs. The external adapters act as API wrappers over APIs, i.e. they make native API calls from within themselves and compute over the results fetched from them. They can be designed using either Python or using JavaScript to work over the fetched details from a publicly authenticated API [17].

At the current stage an external adapter was developed to demonstrate the off-chain computation service. The external adapter fetched weather data from the openweathermap.org's public authenticated API [9]. Temperature data for a city ex. New Delhi was fetched for the last 5 days and was put under statistical analysis. If the current temperature was greater than one half standard deviation from the mean, the result was reported as true i.e. if $Temp > \sigma/2 + Mean$. The simple formula was chosen to notify if extreme weather changes have taken place. It demonstrates how computation can be carried out within an external adapter and can further be extended to carry out data validation for mobile crowd sensing. The mobile crowd sensing application however requires custom data to be fetched which can be achieved by using a custom made public authenticated API. The adapter now needs to be placed on a public marketplace or be hosted by running a chainlink node ourselves, after which it can be called upon from smart contract using its jobID and supplying some LINK fee to the oracle hosting the adapter.

V. Implementation

The following components comprised the development environment:

1.Infura
Infura is a service provider for managed Ethereum nodes, communication to which is done with the help of its API. This saves the time and effort of syncing a local client to set up a full node. It provides a Rinkeby testnet endpoint for testing during development.

2.Metamask
MetaMask provides cryptocurrency wallets used to interact with Ethereum blockchains including both the Mainnet and the testnets.

3.Node.js
Node.js provides the JavaScript environment for Truffle. It was also used for developing an API for running on the localhost, mimicking the data that would be sent for the mobile crowd sensing application.

4.Truffle
Truffle Suite is a development environment based on Ethereum Blockchain, used to develop DApps (Distributed Applications). It can be used for building DApps: Compiling Contracts, Deploying Contracts, Injecting it into a web app, Creating front-end for DApps and Testing. It was especially useful during the process of developing a local blockchain for the 'Blockchain specialization offered by the University at Buffalo'.

5.Ganache
Ganache is part of the TruffleSuite. It provides a local personal blockchain for deploying smart contracts, running applications and carrying out tests.

6.Remix
Remix is an online IDE which was used for solidity development.

Implementation of the sample external adapter was done by following Chainlink's tutorial on the same by Patrick Collins [18]. The final code for the external adapter can be found within the file *index.js* file within the *ExternalAdapter* code folder. Implementation of the calling smart contract that makes the chainlink call for the custom external adapter can be found within the *caller.sol* code file. Also a custom sample API which can be run on the local host was developed and its code can be found within the *index.js* file inside the *CustomApi* folder.

VI. Results

Chainlink provides a spectrum of advantages over traditional smart contracts. By using oracle networks along with smart contracts we get what are called as Hybrid smart contracts i.e. smart contracts which are capable of benefiting from the strengths of both Web2 and Web3. Off-chain computation significantly reduces computation cost. However since the external adapter could not be hosted over a chainlink node deriving exact figures was not possible.

VII. Conclusions

Chainlink is agnostic, and hence can be used with a variety of blockchains that provide the functionality of executing smart contracts. Chainlink offers a network of decentralized oracle networks (DONs), with each DON involving a combination of multiple security techniques needed to service a particular use case. Employing decentralization at the node and data source level ensures no one node or data source acts as a single point of failure, providing users strong guarantees that data will be available, delivered on time, and resistant to manipulation. It provides strict performance check with monetary incentives, due to data signings. Also since the network is immutable, the track records of nodes and data sources remain open for everyone to see and analyze. Providing flexibility for more advanced cryptography (like zero-knowledge proofs) and hardware (such as trusted execution environments) enables oracles to perform additional functions like proving the origin of data and keeping data confidential.

References

[1] B. Warburg and T. Serres, *Basics of Blockchain*, 08 2019.

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Decentralized Business Review*, p. 21260, 2008.

[3] "https://www.section.io/engineering-education/blockchain-consensus-protocols/."

[4] V. Buterin *et al.*, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, 2014.

[5] (2022) What is an oracle in blockchain? explained—chainlink,chain.link. [Online]. Available: https://chain.link/education/blockchain-oracles

[6] (2022) What is an oracle in blockchain? explained — chainlink", chain.link. [Online]. Available: https://blog.chain.link/what-is-oracle-computation/

[7] A. Beniiche, "A study of blockchain oracles," *arXiv preprint arXiv:2004.07140*, 2020.

[8] "What is chainlink? oracles, nodes and LINK tokens," https://www.gemini.com/cryptopedia/what-is-chainlink-and-how-does-it-work, accessed: 2022-5-17.

[9] "Openweathermap api documentation," 2022. [Online]. Available: https://openweathermap.org/guide

[10] "IPFS," https://docs.ipfs.io/concepts/what-is-ipfs/, accessed: 2021-09-15.

[11] J. Huang, L. Kong, H.-N. Dai, L. Cheng, G. Chen, X. Jin, and P. Zeng, "Blockchain based mobile crowd sensing in industrial systems," *IEEE Transactions on Industrial Informatics*, vol. PP, pp. 1–1, 01 2020.

[12] Z. Chen, C. Fiandrino, and B. Kantarci, "On blockchain integration into mobile crowdsensing via smart embedded devices: A comprehensive survey," *Journal of Systems Architecture*, vol. 115, p. 102011, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1383762121000229

[13] X. Liu, R. Chen, Y.-W. Chen, and S.-M. Yuan, "Off-chain data fetching architecture for ethereum smart contract," in *2018 International Conference on Cloud Computing, Big Data and Blockchain (ICCBB)*, 2018, pp. 1–4.

[14] (2022) Chainlink smart contract. [Online]. Available: https://blog.chain.link/smart-contract-use-cases/

[15] (2022) Aavegotchi documentation. [Online]. Available: https://wiki.aavegotchi.com/

[16] F. Zhang, D. Maram, H. Malvai, S. Goldfeder, and A. Juels, "Deco: Liberating web data using decentralized oracles for tls," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, oct 2020. [Online]. Available: https://doi.org/10.1145%2F3372297.3417239

[17] (2022) External adapters. [Online]. Available: https://blog.chain.link/build-and-use-external-adapters/?$_g a$ = 2.162978303.239487839.1652690148 − 473711257.1641551730

[18] (2022) Building and using external adapters. [Online]. Available: https://blog.chain.link/build-and-use-external-adapters/?$_g a$ = 2.162978303.239487839.1652690148 − 473711257.1641551730